

The image is a large, symmetrical, abstract graphic composed of the letters 'S' and 'Y' arranged in a grid-like pattern. The overall shape is a stylized 'Y' or a complex letter 'S'. The top part is a wide horizontal bar made of 'S's, with 'Y's forming a central vertical stem. The sides are also made of 'S's, with 'Y's forming a central vertical stem. The bottom part is a wide horizontal bar made of 'S's, with 'Y's forming a central vertical stem. The entire graphic is composed of these two letters, creating a complex, symmetrical pattern.

```
RRRRRRRR      SSSSSSSS  EEEEEEEEEEE
RRRRRRRR      SSSSSSSS  EEEEEEEEEEE
RR      RR  SS      EE
RR      RR  SS      EE
RR      RR  SS      EE
RR      RR  SS      EE
RRRRRRRR      SSSSSS    EEEEEEEEEEE
RRRRRRRR      SSSSSS    EEEEEEEEEEE
RR  RR      SS      EE
RR  RR      SS      EE
RR  RR      SS      EE
RR  RR      SS      EE
RR      RR  SSSSSSSS  EEEEEEEEEEE
RR      RR  SSSSSSSS  EEEEEEEEEEE
```

....  
....  
....  
....

```
LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```

(1)	40	HISTORY	; DETAILED
(1)	78	DECLARATIONS	
(1)	137	SCHSRSE - REPORT SYSTEM EVENT	
(1)	270	SCH\$UNWAIT - DECREMENT COUNT IN WAIT QUEUE	
(1)	323	SITUATIONAL PRIORITY INCREMENT TABLE	
(1)	342	SCHSCHSE - CHANGE STATE TO EXECUTABLE	
(1)	439	SWPO - SWAP OUT SIMPLE NON-EXECUTABLE	
(1)	457	SCH\$QEND - QUANTUM END ROUTINE	
(1)	612	SENDAST - Send AST to process	
(1)	659	SCH\$WAKE - WAKE PROCESS INTERNAL	
(1)	697	SCH\$SWPWAKE - WAKE SWAPPER PROCESS	



```
0000 1 .TITLE RSE - REPORT SYSTEM EVENT
0000 2 .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6
0000 7 *
0000 8 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 9 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 10 * ALL RIGHTS RESERVED.
0000 11 *
0000 12 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 13 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 14 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 15 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 16 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 17 * TRANSFERRED.
0000 18 *
0000 19 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 20 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 21 * CORPORATION.
0000 22 *
0000 23 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 24 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 25 *
0000 26 *****
0000 27
0000 28 ++
0000 29 FACILITY: EXECUTIVE, SCHEDULER
0000 30
0000 31 ABSTRACT:
0000 32 THIS MODULE CONTAINS THE SYSTEM EVENT REPORTING ROUTINES AND
0000 33 THEIR SUPPORTING SUBROUTINES.
0000 34
0000 35 ENVIRONMENT:
0000 36 MODE = KERNEL
0000 37 --
0000 38
```

```
0000 40 .SBTTL HISTORY ; DETAILED
0000 41
0000 42 :
0000 43 : AUTHOR: R. HUSTVEDT CREATION DATE: 6-SEP-76
0000 44 :
0000 45 : V03-008 SSA0015 Stan Amway 8-Mar-1984
0000 46 : Allow expansion of working set if PFRATH exceeded and
0000 47 : number of active pages (PPG + GPG) exceeds 75% of WSSIZE.
0000 48 : Previously, expansion was done only if the number of active
0000 49 : pages was equal to WSSIZE.
0000 50 : (Acknowledgements go to Wayne Cardoza and Larry Kenah,
0000 51 : who both collaborated on this change.)
0000 52 :
0000 53 : V03-007 WMC0002 Wayne Cardoza 28-Feb-1984
0000 54 : Fix checks for waking swapper.
0000 55 :
0000 56 : V03-006 LY00B4 Larry Yetto 10-FEB-1984 10:29
0000 57 : Fix truncation errors
0000 58 :
0000 59 : V03-005 TMK0002 Todd M. Katz 27-Dec-1983
0000 60 : Fix broken branches.
0000 61 :
0000 62 : V03-004 SSA0003 Stan Amway 5-Dec-1983
0000 63 : Added support for outswap scheduling changes.
0000 64 : Record event time for process unwait and quantum end
0000 65 : event.
0000 66 :
0000 67 : V03-003 TMK0001 Todd M. Katz 13-Nov-1983
0000 68 : Fix broken branches.
0000 69 :
0000 70 : V03-002 TCM0001 Trudy C. Matthews 4-Apr-1983
0000 71 : Change references to working set fields in PHD so that
0000 72 : they are used as unsigned words.
0000 73 :
0000 74 : V03-001 WMC0001 Wayne Cardoza 11-Mar-1983
0000 75 : Bad comparison against BORROWLIM.
0000 76 :
```



```
0000 78          .SBTTL  DECLARATIONS
0000 79
0000 80  :
0000 81  : INCLUDE FILES:
0000 82  :
0000 83          $ACBDEF      : DEFINE AST CONTROL BLOCK
0000 84          $DYNDEF      : DEFINE STRUCTURE TYPE CODES
0000 85          $CEBDEF      : DEFINE COMMON EVENT BLOCK
0000 86          $IPLDEF      : IPL DEFINITIONS
0000 87          $PCBDEF      : PCB DEFINITIONS
0000 88          $PHDDEF      : PROCESS HEADER DEFINITIONS
0000 89          $PRDEF       : PROCESSOR REGISTER DEFS
0000 90          $PRIDEF      : PRIORITY INCREMENT CLASSES
0000 91          $SSDEF       : DEFINE STATUS CODES
0000 92          $STATEDEF    : STATE DEFINITIONS
0000 93          $WQHDEF      : WAIT QUEUE HEADER DEFINITIONS
0000 94  :
0000 95  : MACROS:
0000 96  :
0000 97          .MACRO  EVENT,EVTN,STATLIST,EACTION,CONT=0
0000 98          .IF     NB,EVTN
0000 99  EVT$_'EVTN==EVTCTR
0000 100         .ENDC
0000 101  EVTCTR=EVTCTR+1
0000 102         .WORD   EACTION-STACT
0000 103  RSE...=.
0000 104         .PSECT  AES2,BYTE
0000 105  STMSK=CONT
0000 106         .IRP    ST,<STATLIST>
0000 107  STMSK=STMSK+<1@SCH$C_'ST>
0000 108         .ENDR
0000 109         .LONG   STMSK
0000 110         .PSECT  AES1,BYTE
0000 111  .=RSE...
0000 112         .ENDM   EVENT
0000 113  :
0000 114  : GENERATE MASK FOR WAIT STATES
0000 115  :
0000 116  : GMASK  STATENAME
0000 117  :
0000 118  :
0000 119  :
0000 120         .MACRO  GMASK,STATE
0000 121  ST=CH$C_'STATE
0000 122  WAITST=WAITST+<1@ST>
0000 123         .ENDM   GMASK
0000 124  :
0000 125  : EQUATED SYMBOLS:
0000 126  :
00000000 0000 127  EVTCTR=0      : INITIALIZE EVENT COUNTER TO 0
00000000 0000 128  WAITST=0     : INITIALIZE WAIT STATE MASK
00000000 0000 129  ASTEXIT=0    : AST EXIT CHANGE MODE CODE
0000 130  :
0000 131  : OWN STORAGE:
0000 132  :
00000000 0000 133         .PSECT  AES2,BYTE      : STATE EVENT MASK PSECT
00000000 0000 134  STET=      : BASE OF STATE EVENT TABLE
```

RSE  
V04-000

- REPORT SYSTEM EVENT  
DECLARATIONS

1 2

16-SEP-1984 01:06:34 VAX/VMS Macro V04-00  
5-SEP-1984 03:47:04 [SYS.SRC]RSE.MAR;1

Page 4  
(1)

00000000 135

.PSECT AES1,BYTE

```
0000 137 .SBTTL SCH$RSE - REPORT SYSTEM EVENT
0000 138
0000 139 :++
0000 140 : FUNCTIONAL DESCRIPTION:
0000 141 : SCH$RSE RECEIVES SYSTEM EVENT REPORTS FROM VARIOUS SOURCES
0000 142 : AND PERFORMS THE APPROPRIATE ACTION FOR THE SPECIFIED PROCESS.
0000 143 : EVENT REPORTING MUST BE PERFORMED WITH IPL=IPL$ SYNCH.
0000 144 : AS A SIDE EFFECT OF AN EVENT REPORT, THE RESCHEDULING INTERRUPT
0000 145 : MAY BE TRIGGERED IF APPROPRIATE.
0000 146
0000 147 : CALLING SEQUENCE:
0000 148 : BSB/JSB SCH$RSE
0000 149 : .BYTE EVT$_EVENTNAME
0000 150
0000 151 : THIS CALLING SEQUENCE IS GENERATED BY THE RPTEVT SYSTEM MACRO
0000 152
0000 153 : REPEVT EVENTNAME
0000 154
0000 155 : INPUT PARAMETERS:
0000 156 : R2 - SITUATIONAL PRIORITY INCREMENT CLASS NUMBER
0000 157 : R4 - PCB ADDRESS OF PROCESS FOR WHICH EVENT IS REPORTED
0000 158
0000 159 : EVENT NUMBER CONTAINED IN BYTE LOCATED BY ADDRESS AT TOP
0000 160 : OF STACK. @ (SP)
0000 161
0000 162 : IMPLICIT INPUTS:
0000 163 : SCHEDULER DATA BASE
0000 164
0000 165 : OUTPUT PARAMETERS:
0000 166 : NONE
0000 167
0000 168 : IMPLICIT OUTPUTS:
0000 169 : NONE
0000 170
0000 171 : COMPLETION CODES:
0000 172 : NONE
0000 173
0000 174 : SIDE EFFECTS:
0000 175 : A RESECHEDULING INTERRUPT MAY BE REQUESTED IF THE SPECIFIED
0000 176 : PROCESS IS HIGHER IN PRIORITY THAN THE CURRENT PROCESS.
0000 177
0000 178 :--
0000 179
0000 180 SCH$RSE::
0000 181 : REPORT SYSTEM EVENT
0000 182 : GET EVENT NUMBER
0000 183 : UPDATE RETURN ADDRESS
0000 184 : GET CURRENT STATE NUMBER
0000 185 : GET STATE MASK FOR EVENT
0000 186 : DO ACTION IF STATE BIT SET
0000 187 : CHECK NEXT ACTION
0000 188 : IF CONTINUATION
0000 189 : OTHERWISE IGNORE EVENT
0000 190 :
0000 191 :
0000 192 :
0000 193 :
53 00 BE 9A 0000 181 MOVZBL @ (SP),R3 : REPORT SYSTEM EVENT
51 2C A4 3C 0004 182 INCL (SP) : GET EVENT NUMBER
50 0000 CF 43 D0 0006 183 MOVZWL PCB$W STATE(R4),R1 : UPDATE RETURN ADDRESS
06 50 51 E0 000A 184 10$: MOVL W^STET[R3],R0 : GET CURRENT STATE NUMBER
53 D6 0014 185 BBS R1,R0,ACTION : GET STATE MASK FOR EVENT
F1 50 E8 0016 186 INCL R3 : DO ACTION IF STATE BIT SET
0A' 00 53 CF 001A 187 BLBS R0,10$ : CHECK NEXT ACTION
001E 188 RSB : IF CONTINUATION
001E 189 : OTHERWISE IGNORE EVENT
001E 190 ACTION: CASEL R3,#0,S^#MAXEVT : SWITCH ON EVENT NUMBER(UPDATED)
001E 191 STACT: : BASE OF ACTION TABLE
001E 192 EVENT AST,<- : AST EVENT
001E 193 CEF,- : COMMON EVENT FLAG WAIT
```



```
00000000 001E 194 COLPG,- ; COLLIDED PAGE WAIT
          001E 195 FPG,- ; FREE PAGE WAIT
          001E 196 HIB,- ; RESIDENT HIBERNATE
          001E 197 HIBO,- ; NON-RESIDENT HIBERNATE
          001E 198 LEF,- ; LOCAL EVENT FLAG WAIT
          001E 199 LEFO,- ; LOCAL EVENT FLAG WAIT (NON-RES)
          001E 200 MWAIF,- ; MUTEX WAIT
          001E 201 PFW,- ; PAGE FAULT WAIT
          001E 202 >,EVENTE ; AST EXECUTABLE STATE CHANGE
          0020 203 EVTS,_COLPGA==EVTS,_AST ; USE SAME EVENT FOR COLLIDED PAGE AVAIL
          0020 204
          0020 205
          0020 206 EVENT EVENT,<LEF>,LEFEVT,CONT=1 ; EVENT FLAG SETTING
          0022 207 EVENT <CEF>,CEFEVT,CONT=1 ; COMMON EVENT FLAG SET
          0024 208 EVENT <LEFO>,EVENTE ; NON-RESIDENT LOCAL EVENT
          0026 209
          0026 210 EVENT FPG,- ; FREE PAGE AVAILABLE
          0026 211 FPG,- ; FREE PAGE WAIT STATE
          0026 212 >,EVENTF ; EXECUTABLE STATE CHANGE
          0028 213 EVENT WAKE,<- ; WAKE EVENT
          0028 214 HIB,- ; RESIDENT HIBERNATE
          0028 215 HIBO,- ; NON-RESIDENT HIBERNATE
          0028 216 >,EVENTE ;
          002A 217
          002A 218
          002A 219 EVENT RESUME,<- ; RESUME EVENT
          002A 220 SUSP,- ; RESIDENT SUSPENDED
          002A 221 SUSPO,- ; NON-RESIDENT SUSPENDED
          002A 222 >,EVENTE ; EXECUTABLE
          002C 223
          002C 224
          002C 225 EVENT PFCOM,<- ; PAGE FAULT COMPLETE EVENT
          002C 226 PFW,- ; PAGE FAULT WAIT
          002C 227 >,EVENTE ; EXECUTABLE
          002E 228
          002E 229 EVENT SETPRI,<- ; SET PRIORITY EVENT
          002E 230 COM,- ; RESIDENT COMPUTE
          002E 231 COMO,- ; NON-RESIDENT COMPUTE
          002E 232 >,EVENTF ; EXECUTABLE
          0030 233
          0030 234 EVENT SWPOUT,<- ; SWAP OUT EVENT
          0030 235 HIB,- ; RESIDENT HIBERNATE
          0030 236 LEF,- ; RESIDENT LOCAL EVENT FLAG WAIT
          0030 237 SUSP,- ; RESIDENT SUSPENDED
          0030 238 >,SWPO,CONT=1 ; SIMPLE SWAP OUT
          0032 239 EVENT <- ; SWAP OUT EVENT CONINUATION
          0032 240 COM,- ; RESIDENT COMPUTE
          0032 241 >,SWPOE ; EXECUTABLE OUTSWAP
          0034 242
          0034 243 :
          0034 244 : IF CASE FALLS THROUGH, THEN BUGCHECK WITH ILLEGAL EVENT
          0034 245 : NUMBER.
          0034 246 :
          0000000A 0034 247 MAXEVT=EVTCTR-1 ; MAXIMUM EVENT NUMBER
          0034 248 BUG_CHECK ILLEVNUM,FATAL ; ILLEGAL EVENT NUMBER
          0038 249
```

```
0038 251 :  
0038 252 : COMMON EVENT FLAG SET  
0038 253 :  
0038 254 CEF EVT: :  
OE 24 A4 E9 0038 255 BLBC PCB$ _STS(R4),EVENTE : BR IF NOT RESIDENT  
53 6C A4 DO 003C 256 LEF EVT: : LOCAL EVENT FLAG SET FOR RESIDENT PROCESS  
00C0 C3 04 CO 003C 257 MOVL PCB$ _PHD(R4),R3 : POINT TO PHD  
0088 C3 01 3C 0040 258 ADDL #4,PHD$ _PC(R3) : SKIP PC OVER CHMK INSTRUCTION  
0045 259 MOVZWL #SS$ _NORMAL,PHD$ _R0(R3); SET NORMAL COMPLETION FOR WAIT  
004A 260 :  
004A 261 : EVENT EXECUTABLE ACTION ROUTINE  
004A 262 :  
004A 263 :  
004A 264 EVENTE: : EVENT EXECUTABLE STATE CHANGE  
88'AF 9F 004A 265 EVENTF: : ACTIVATE WITH NO WAIT TIME ACCOUNTING  
004D 266 PUSHAB B* SCH$CHSE : MAKE UNWAIT EXIT THROUGH CHSE  
004D 267 ; BRB SCH$UNWAIT : AND FALL INTO UNWAIT  
004D 268
```

```
004D 270 .SBTTL SCH$UNWAIT - DECREMENT COUNT IN WAIT QUEUE
004D 271 :++
004D 272 :
004D 273 FUNCTIONAL DESCRIPTION:
004D 274 SCH$UNWAIT DECREASES THE NUMBER OF PROCESSES IN THE WAIT
004D 275 QUEUE SELECTED BY THE SPECIFIED PCB AND STATE VALUE.
004D 276 :
004D 277 CALLING SEQUENCE:
004D 278 BSB/JSB SCH$UNWAIT
004D 279 :
004D 280 INPUT PARAMETERS:
004D 281 R1 - STATE NUMBER (PRESERVED)
004D 282 R2 - UNUSED (PRESERVED)
004D 283 R4 - PCB ADDRESS (PRESERVED)
004D 284 :
004D 285 IMPLICIT INPUTS:
004D 286 PCB LOCATED BY ADDRESS IN R4
004D 287 :
004D 288 IMPLICIT OUTPUTS:
004D 289 COUNT IN WAIT QUEUE HEADER IS DECREMENTED IF STATE IS A WAIT
004D 290 STATE.
004D 291 :
004D 292 :--
004D 293 :
004D 294 SCH$UNWAIT::
1A 7B'AF 51 E1 004D 295 BBC R1,B^WAITMSK,20$ : DECREMENT PROPER WAIT COUNT
51 03 B1 0052 296 CMPW #SCH$C_CEF,R1 : SKIP OUT IF NOT WAIT STATE
16 13 0055 297 BEQL 30$ : CHECK FOR COMMON EVENT FLAG WAIT
51 0C C4 0057 298 MULL #WQHSC_LENGTH,R1 : COMPUTE BYTE INDEX TO WQ HDR
50 00000000'EF41 9E 005A 299 MOVAB L^SCH$AQ_WQHDR[R1],R0 : COMPUTE ADDRESS OF WAIT Q HEADER
08 A0 B7 0062 300 10$: DECH WQH$W_WQNT(R0) : DECREMENT WAIT QUEUE COUNT
0118 C4 0000'CF D0 0065 301 MOVL W^EXESGL_ABSTIM,PCBSL_WAITTIME(R4) : Record event time
05 006C 302 20$: RSB : RETURN
006D 303 :
50 2E A4 9A 006D 304 30$: MOVZBL PCBSB_WEFC(R4),R0 : WAIT CLUSTER NUMBER
50 50 A440 D0 0071 305 MOVL PCBSL_EFCS(R4)[R0],R0 : GET CLUSTER ADDRESS
50 14 C0 0076 306 ADDL #CEBSL_WQFL,R0 : POINT TO WAIT QUEUE HEADER
E7 11 0079 307 BRB 10$ : GO DECREMENT WAIT COUNT
007B 308 :
007B 309 GMASK CEF : COMMON EVENT FLAG
007B 310 GMASK LEF : LOCAL EVENT FLAG WAIT
007B 311 GMASK LEFO : LOCAL EVENT FLAG WAIT
007B 312 GMASK HIB : HIBERNAT WAIT
007B 313 GMASK HIBO : HIBERNATE WAIT
007B 314 GMASK FPG : FREE PAGE WAIT
007B 315 GMASK COLPG : COLLISION PAGE WAIT
007B 316 GMASK PFU : PAGE FAULT WAIT
007B 317 GMASK SUSP : SUSPENDED WAIT
007B 318 GMASK SUSPO : SUSPENDED WAIT
007B 319 GMASK MWAIT : MUTEX WAIT
00000FFE 007B 320 WAITMSK:.LONG : MASK OF WAIT STATES
007F 321 WAITST
```



```

007F 323 .SBTTL SITUATIONAL PRIORITY INCREMENT TABLE
007F 324
007F 325 : FIXED DATA:
007F 326 : SITUATIONAL PRIORITY INCREMENT TABLE
007F 327 : (INDEXED BY PRIORITY INCREMENT CLASS)
007F 328 :
007F 329 :
007F 330 B_PINC:
00 007F 331 .BYTE 0 : CLASS 0 - NONE
02 0080 332 .BYTE 2 : CLASS 1 - I/O COMPLETE
03 0081 333 .BYTE 3 : CLASS 2 - RESOURCE AVAIL
04 0082 334 .BYTE 4 : CLASS 3 - TERM OUTPUT COMP
06 0083 335 .BYTE 6 : CLASS 4 - TERM INPUT COMP
0084 336
0084 337
0084 338 EXESTATE: : EXECUTABLE STATE MASK
00003000 0084 339 .LONG <1@SCH$C_COM>!<1@SCH$C_COMO>
0088 340

```

```
0088 342 .SBTTL SCH$CHSE - CHANGE STATE TO EXECUTABLE
0088 343
0088 344 :++
0088 345 FUNCTIONAL DESCRIPTION:
0088 346 SCH$CHSE CHANGES THE STATE OF A PROCESS, AS REPRESENTED BY
0088 347 ITS PCB, TO AN EXECUTABLE STATE. THE RESCHEDULING INTERRUPT
0088 348 WILL BE TRIGGERED IF THE PROCESS IS RESIDENT AND HAS A PRIORITY
0088 349 GREATER THAN THAT OF THE CURRENTLY EXECUTING PROCESS. A
0088 350 PRIORITY INCREMENT CLASS NUMBER SUPPLIED AS A REGISTER CONTAINED
0088 351 ARGUMENT IS USED TO COMPUTE THE NEW PROCESS PRIORITY FROM ITS
0088 352 BASE PRIORITY.
0088 353
0088 354 CALLING SEQUENCE:
0088 355 BSB/JSB SCH$CHSE
0088 356
0088 357 INPUT PARAMETERS:
0088 358 R0 - NEW PRIORITY (SCH$CHSEP ONLY)
0088 359 R2 - PRIORITY INCREMENT CLASS NUMBER (SCH$CHSF ONLY)
0088 360 0 => NO INCREMENT (PAGEFAULT I/O COMPLETION)
0088 361 1 => NON-TERMINAL I/O COMPLETION
0088 362 2 => RESOURCE AVAILABILITY
0088 363 3 => TERMINAL OUTPUT COMPLETION
0088 364 4 => TERMINAL INPUT COMPLETION
0088 365 R4 - PCB ADDRESS
0088 366
0088 367 IMPLICIT INPUTS:
0088 368 SCH$AQ_COMT - COMPUTE QUEUE HEADERS FOR COM,COMO STATES
0088 369 SCH$GB_PRI - CURRENT PROCESS PRIORITY.
0088 370
0088 371 OUTPUT PARAMETERS:
0088 372 R2 - R2, PRIORITY INCREMENT CLASS NUMBER IF SCH$CHSE. (PRESERVED)
0088 373 R3 - R3 (PRESERVED)
0088 374
0088 375 IMPLICIT OUTPUTS:
0088 376 SCH$AQ_COMH - VECTOR OF COMPUTE QUEUE HEADERS.
0088 377 SCH$GL_COMQS - COMPUTE QUEUE SUMMARY BIT VECTOR.
0088 378
0088 379 COMPLETION CODES:
0088 380 NONE
0088 381
0088 382 SIDE EFFECTS:
0088 383 THE PCB SPECIFIED IS REMOVED FROM ITS PRESENT STATE QUEUE
0088 384 AND INSERTED IN THE APPROPRIATE COMPUTE QUEUE, COM OR COMO.
0088 385 AT THE PRIORITY COMPUTED FOR THE SPECIFIED SITUATION CLASS.
0088 386 THE SUMMARY BIT FOR THE DESTINATION STATE QUEUE IS SET TO
0088 387 NOTE THAT IT IS OCCUPIED.
0088 388 IF THE NEW PRIORITY FOR THE PROCESS IS GREATER THAN THAT OF
0088 389 CURRENT PROCESS AND IT IS RESIDENT, THE RESCHEDULING INTERRUPT
0088 390 WILL BE TRIGGERED.
0088 391
0088 392 --
0088 393 SCH$CHSE::
0088 394 CLRL R0 ;CHANGE TO EXECUTABLE STATE
0088 395 SUBB3 B,PINC[R2],PCBSB_PRI[R4] ;CLEAR HIGH SUM BITS FOR ADDB
0088 396 CMPB R0,PCBSB_PRI[R4] ;ADD PRIORITY INCR
0088 397 BLEQ 10$ ;CHECK FOR > CURRENT PRI
0088 398 MOVB PCBSB_PRI[R4],R0 ;NO
; KEEP CURRENT PRIORITY INSTEAD
```

```
50 2F A4 F1 AF42 50 D4 0088
0B A4 50 83 008A
04 91 0091
50 0B A4 15 0095
04 90 0097
```

```
10 50 91 009B 399 10$: CMPB R0,#16      : CHECK FOR RESULT >15
      04 18 009E 400      BGEQ SCH$CHSEP    : YES, USE COMPUTED VALUE
50 2F A4 90 00A0 401      MOVB PC$B_PRI$B(R4),R0 : KEEP AT BASE IF LESS
      00A4 402
      00A4 403
      00A4 404
      00A4 405
      00A4 406
      00A4 407 SCH$CHSEP::
      00A4 408 MFPR #PR$ IPL,R1      : ENTRY WITH PRIO IN R0
      08 51 D1 00A7 409      CMPL R1,#IPL$_SYNCH : GET IPL
      62 19 00AA 410      BLSS BADIPL          : MUST BE AT SYNCH OR GREATER
      51 64 0F 00AC 411      REMQUE (R4),R1     : NO, FATAL ERROR
      1C 12 00AF 412      BNEQ 10$            : REMOVE FROM CURREN QUEUE
      51 2C A4 3C 00B1 413      MOVZWL PC$W_STATE(R4),R1 : CONTINUE IF QUEUE NOTEMPTY
13 CB AF 51 E1 00B5 414      BBC R1,EXESTATE,10$ : GET OLD STATE
      51 0B A4 9A 00BA 415      MOVZBL PC$B_PRI$(R4),R1 : NO SUMMARY BITS
      03 2C A4 E9 00BE 416      BLBC PC$W_STATE(R4),5$ : GET CURRENT PRI
      51 20 C0 00C2 417      ADDL #32,R1        : SKIP IF RESIDENT
00 00000000'EF 51 E5 00C5 418 5$: BBCC R1,L^SCH$GL_COMQS,10$ : MAKE NONRES PRIO
      0B A4 50 90 00CD 419 10$: MOVB R0,PC$B_PRI$(R4) : CLEAR PRESENCE BIT FOR STATE
      51 0C D0 00D1 420      MOVL #SCH$C_COM,R1 : SAVE NEW PRIO
      12 24 A4 E8 00D4 421      BLBS PC$B_L_STS(R4),20$ : ASSUME COM STATE
      51 51 D6 00D8 422      INCL R1           : CHECK FOR RESIDENCE
00 00000000'EF 50 C0 00DA 423      ADDL2 #32,R0 : COMO=COM+1
      01CD 30 E2 00DD 424      BBSS R0,L^SCH$GL_COMQS,15$ : COMO HEADERS FOLLOW COM
      14 11 00E5 425 15$: BSBW SCH$SWPWAKE : SET SUMMARY BIT FOR NEW QUEUE
      50 00000000'EF 91 00EA 426      BRB 35$ : WAKE SWAPPER
      03 19 00F1 427 20$: CMPB L^SCH$GB_PRI,R0 : COMPLETE STATE CHANGE
      00F3 428      BLSS 30$ : IS PRIO GREATER THAN CURRENT PROCESS
      00F6 429      SOFTINT #IPL$ SCHED : NO, DONT RESCHEDULE
      2C A4 51 B0 00FE 430 30$: BBSS R0,L^SCH$GL_COMQS,35$ : TRIGGER RESCHEDULE INTERRUPT
51 00000000'EF 40 7E 0102 431 35$: MOVW R1,PC$W_STATE(R4) : SET SUMMARY BIT FOR NEW QUEUE
      91 64 0E 010A 432      MOVAQ L^SCH$AQ_COMT[R0],R1 : SET NEW STATE
      05 010D 433      INSQUE (R4),@ (RT)+ : COMPUTE HDR ADDR
      010E 434      RSB : INSERT IN NEW QUEUE
      010E 435      : RETURN
      0112 436 BADIPL: BUG_CHECK BADRSEIPL,FATAL : BAD IPL AT ENTRANCE TO RSE
      0112 437
```



```
0112 439 .SBTTL SWPO - SWAP OUT SIMPLE NON-EXECUTABLE
0112 440
0112 441 :
0112 442 :
0112 443 SWPO: : NON-EXECUTABLE OUTSWAP
FF38 30 0112 444 BSBW SCH$UNWAIT : REMOVE FROM WAIT QUEUE
2C A4 B6 0113 445 INCW PCBSW_STATE(R4) : UPDATE STATE NUMBER
51 64 OF 0118 446 REMQUE (R4),R1 : REMOVE FROM WAIT QUEUE
10 B0 64 OE 011B 447 INSQUE (R4),@WQH$SL WQBL+WQH$C_LENGTH(R0) : INSERT AT TAIL OF QUEUE
14 A0 B6 011F 448 INCW WQH$W_WQCNT+WQH$C_LENGTH(R0) : NOTE COUNT IN WAIT QUEUE
OS 05 0122 449 RSB : EXIT
0123 450
0123 451 :
0123 452 :
0123 453 :
50 0B A4 9A 0123 454 SWPOE: MOVZBL PCBSB_PRI(R4),R0 : GET PRIORITY
FF7A 31 0127 455 BRW SCH$CRSEP : AND CHANGE TO COMO
```

```
012A 457 .SBTTL SCH$QEND - QUANTUM END ROUTINE
012A 458
012A 459 :++
012A 460
012A 461 FUNCTIONAL DESCRIPTION:
012A 462 SCH$QEND IS CALLED BY THE TIMER WHEN THE QUANTUM FOR THE CURRENT
012A 463 PROCESS HAS BEEN EXHAUSTED. A NEW QUANTUM IS INITIALIZED
012A 464 THE PROCESS PLACED AT ITS BASE PRIORITY AND THE RESCHEDULING
012A 465 INTERRUPT TRIGGERED. A CHECK IS MADE FOR CPU TIME LIMIT EXPIRATION
012A 466 AND APPROPRIATE EXIT ASTS GENERATED WHEN THE LIMIT IS REACHED.
012A 467 THE AUTOMATIC WORKING SET SIZE LOGIC IS INVOKED IF ENABLED TO
012A 468 TRADEOFF WORKING SET SIZE AGAINST PAGEFAULT RATE.
012A 469
012A 470 CALLING SEQUENCE:
012A 471 BSB/JSB SCH$QEND
012A 472
012A 473 INPUT PARAMETERS:
012A 474 R4 - PCB ADDRESS OF CURRENT PROCESS
012A 475 R5 - PROCESS HEADER ADDRESS
012A 476
012A 477 IMPLICIT INPUTS:
012A 478 PCB OF CURRENT PROCESS
012A 479 PROCESS HEADER OF CURRENT PROCESS
012A 480
012A 481 IMPLICIT OUTPUTS:
012A 482 PHDSW_QUANT - INITIALIZED TO A NEW QUANTUM
012A 483 PCBSV_INQUAN - INITIAL QUANTUM FLAG CLEARED
012A 484
012A 485 :--
012A 486
012A 487 SCH$QEND::: QUANTUM END ROUTINE
012A 488 BBCC #PCBSV_INQUAN,PCBSL_STS(R4),10$ : CLEAR INITIAL QUAN FLAG
3C A5 00 24 A4 03 E5 012A 489 10$: MOVW SCH$GW_QUAN,PHDSW_QUANT(R5) : SET NEW QUANTUM
0118 C4 0000'CF D0 012F 490 MOVL W^EXESGL_ABSTIM,PCBSL_WAITIME(R4) : Record event time
10 0B A4 91 013E 491 CMPB PCBSB_PRI(R4),#16 : CHECK FOR REAL-TIME
26 19 0142 492 BLSS 50$ : YES
0144 493 :
0144 494 :
0144 495 : CHECK FOR CPU TIME LIMIT EXPIRATION
0144 496 :
0144 497 TSTL PHDSL_CPULIM(R5) : IS THERE ANY LIMIT?
0147 498 BNEQ 60$ : YES, GO CHECK IT OUT
09 24 A4 18 E0 0149 499 40$: BBS #PCBSV_DISAWS,PCBSL_STS(R4),45$ : BRANCH IF ADJUSTMENT DISABLED
53 0000'CF D0 014E 500 MOVL W^SCH$GL_USINC,R3 : ASSUME INCREMENT
02 13 0153 501 BEQL 45$ : BR IF NO AUTO WS ADJUSTMENT
49 10 0155 502 BSBB WSADJUST : ELSE GO DO IT
00000000'EF D5 0157 503 45$: TSTL L^SCH$GL_COMOQS : IS THERE ANY INSWAP PENDING?
0B A4 2F A4 90 015D 504 BEQL 47$ : NO
014E 30 015F 505 MOVB PCBSB_PRI(R4),PCBSB_PRI(R4) : YES, FORCE TO BASE PRIORITY
05 0164 506 BSBW SCH$SWP_WAKE : AND WAKE SWAPPER
0167 507 47$: SOFTINT #IPL$_SCHED : TRIGGER RESCHEDULING INT
016A 508 50$: RSB : AND RETURN
016B 509
016B 510 :
016B 511 :
016B 512 : A non-zero limit exists, check for processor time expiration
016B 513 : If CPU time limit is exceeded then an additional amount of time will
```

```
016B 514 : be allowed for each access mode. An AST will be issued to cause an
016B 515 : exit for each of the access modes. The additional time allowance will
016B 516 : be provided for each access mode.
016B 517 :
016B 518 :
50 38 A5 5C A5 C3 016B 519 60$: SUBL3 PHD$$_CPULIM(R5),PHD$$_CPUTIM(R5),R0 : HAS LIMIT BEEN REACHED
D6 1F 0171 520 BLSSU 40$ : NO, CONTINUE NORMALLY
0173 521 :
0173 522 : CPU LIMIT HAS EXPIRED, AN AST WILL BE SENT TO NOTIFY THE PROCESS
0173 523 :
50 0000'CF C0 0173 524 ADDL2 W^SGN$GL_EXTRACPU,R0 : COMPUTE TOTAL AMOUNT OF EXTRA TIME
5C A5 50 C0 0178 525 ADDL2 R0,PHD$$_CPULIM(R5) : GIVE EXTRA TIME FOR CLEANUP
0110 C5 50 C0 017C 526 ADDL2 R0,PHD$$_EXTRACPU(R5) : AND RECORD AMOUNT OF EXTRA TIME
50 60 A5 9E 0181 527 MOVAB PHD$$_CPOMODE(R5),R0 : GET ADDRESS OF AST ACCESS MODE
C1 AF 9F 0185 528 PUSHAB 40$ : SET RETURN ADDRESS
53 20AC 8F 3C 0188 529 MOVZWL #SS$$_EXCPUTIM,R3 : PASS EXIT STATUS TO SENDAST
018D 530 SCH$FORCEDEXIT::
018D 531 BSBW SENDAST : SEND AST TO PROCESS
0190 532 :
0190 533 : CPU TIME EXPIRATION AST HANDLER
0190 534 :
00 0000 0190 535 (PUABRT: WORD 0 : NULL ENTRY MASK
BC 0192 536 CHMK S^#ASTEXIT : EXIT FROM AST ROUTINE (CLEAR AST)
F4 11 0194 537 10$: $EXIT_S 4(AP) : EXIT TO INVOKE EXIT HANDLERS
019E 538 BRB 10$ : JUST IN CASE
01A0 539 :
01A0 540 : Adjust working set size automatically to achieve desired tradeoff
01A0 541 : between page fault rate and working set size. There are two page
01A0 542 : fault rate thresholds: SCH$GL_PFRATL, the lower threshold and
01A0 543 : SCH$GL_PFRATH, the higher threshold. Each time SCH$QEND is invoked,
01A0 544 : the page fault rate is computed and compared with these thresholds.
01A0 545 : If it is above the high threshold the working set size is increased
01A0 546 : by SCH$GW_WSINC and if the rate is below the lower threshold, the
01A0 547 : working set size is decreased by SCH$GW_WSEDEC. The actual adjustment
01A0 548 : is performed by a normal kernel mode AST.
01A0 549 :
01A0 550 : Automatic adjustment of working set size is constrained by the values:
01A0 551 : SCH$GW_AWSMIN and WSEXTENT per process that establish upper and lower
01A0 552 : values for automatic working set size adjustment. Working set size
01A0 553 : adjustment is further constrained by the process quota.
01A0 554 :
01A0 555 :
01A0 556 :
01A0 557 : R3 - Working set increment
01A0 558 :
01A0 559 WSADJUST:
50 0100 C5 C3 01A0 560 SUBL3 PHD$$_TIMREF(R5),- : AUTO-ADJUST WORKING SET SIZE
38 A5 02 12 01A4 561 PHD$$_CPUTIM(R5),R0 : COMPUTE DELTA-T
02 50 D6 01A7 562 BNEQ 10$ : BR IF NON-ZERO
01A9 563 INCL R0 : ELSE FORCE TO ONE FOR DIVIDE
01AB 564 10$:
0000'CF 50 D1 01AB 565 CMPL R0,W^SCH$GL_AWSTIME : IS THIS A MEANINGFUL INTERVAL?
3F 19 01B0 566 BLSS NOADJUST : NO, TRY AGAIN LATER
00FC C5 C3 01B2 567 SUBL3 PHD$$_PFLREF(R5),- : COMPUTE DELTA-PGFLT
51 4C A5 01B6 568 PHD$$_PAGEFLTS(R5),R1 :
00FC C5 4C A5 D0 01B9 569 MOVL PHD$$_PAGEFLTS(R5),PHD$$_PFLREF(R5) : SAVE NEW PAGE FAULT REF
0100 C5 38 A5 D0 01BF 570 MOVL PHD$$_CPUTIM(R5),PHD$$_TIMREF(R5) : AND SAVE CPUTIME REF
```



```
51 000003E8 8F C4 01C5 571 MULL #1000,R1 ; MULTIPLY BY SCALE FACTOR
      51 50 C6 01CC 572 DIVL R0,R1 ; AND COMPUTE PAGEFLTS/10SEC
      00F8 C5 51 D0 01CF 573 MOVL R1,PHD$PFLTRATE(R5) ; SAVE CURRENT RATE
      0000 CF 51 D1 01D4 574 CMPL R1,W^SCH$GL_PFRATH ; ARE WE ABOVE HIGH THRESHOLD?
      53 0000 CF 17 18 01D9 575 BGEQ ADJUSTUP ; YES,
      0000 CF 51 CE 01DB 576 MNEGL W^SCH$GL_WSDEC,R3 ; NO, GET DECREMENT VALUE
      0000 CF 51 D1 01E0 577 CMPL R1,W^SCH$GL_PFRATH ; ARE WE BELOW LOW THRESHOLD?
      0000 CF 36 0A 18 01E5 578 BGEQ NOADJUST ; NO, IN DEAD BAND -- NOTHING TO DO
      02 18 01ED 580 CMPW PCB$W_PPGCNT(R4),W^SCH$GL_WSDEC ; ARE WE AT LOWER WS LIMIT?
      39 11 01EF 581 BLEQU NOADJUST ; YES, NOTHING TO DO
      05 01F1 582 NOADJUST:
      05 01F1 583 RSB
      05 01F2 584 ADJUSTUP:
      51 18 A5 08 A5 A3 01F2 585 SUBW3 PHD$W_WSLIST(R5),PHD$W_WSQUOTA(R5),R1
      50 50 A5 3C 01F8 586 ; ASSUME HIGH LIMIT WILL BE QUOTA
      00000000 EF 0000 CF D1 01FC 587 MOVZWL PHD$W_WSSIZE(R5),R0 ; GET CURRENT WORKING SET SIZE
      06 1A 0205 588 CMPL W^SCH$GL_BORROWLIM,L^SCH$GL_FREECNT ; ARE THERE LOTS OF FREE PAGES?
      51 16 A5 08 A5 A3 0207 589 BGTRU 10$ ; BRANCH IF MEMORY IS AT A PREMIUM
      51 50 B1 020D 590 SUBW3 PHD$W_WSLIST(R5),PHD$W_WSEXTENT(R5),R1
      51 50 B1 020D 591 ; ALLOW LARGER GROWTH SIZE
      51 50 B1 0210 592 10$: CMPW R0,R1 ; ARE WE AT MAXIMUM SIZE?
      51 36 A4 34 A4 A1 0212 593 BGTRU NOADJUST ; YES, CAN'T GO ANY LARGER
      51 50 B1 0218 594 ADDW3 PCB$W_GPGCNT(R4),PCB$W_PPGCNT(R4),R1 ; GET CURRENT PHYSICAL SIZE
      52 50 FE 8F 78 021B 595 CMPW R0,R1 ; Be sure that pages in use don't exceed WS
      50 52 A2 0222 596 BLSSU WSERR ; BRANCH IF WS SMALLER THAN PAGES IN USE
      50 51 B1 0225 597 ASHL #2,R0,R2 ; Compute 75% of WSSIZE as page threshold
      50 51 B1 0225 598 SUBW2 R2,R0
      50 51 B1 0225 599 CMPW R1,R0
      50 61 A5 9E 0228 600 BLSSU NOADJUST ; If threshold not exceeded,
      1D 10 022E 601 ADJUST: MOVAB PHD$B_AWSMODE(R5),R0 ; skip WS adjustment
      0230 602 BSBB SENDAST ; GET ADDRESS OF AST ACCESS MODE
      0230 603 ; SEND AST TO PROCESS
      51 00000000 9F 0000 0230 604 ADJWS: .WORD D
      61 A1 94 0232 605 MOVL @#CTL$GL_PHD,R1 ; GET PHD ADDRESS SO
      04 0239 606 CLRB PHD$B_AWSMODE(R1) ; ACCESS MODE FLAG CAN BE RESET
      0248 607 $ADJWSL_S 4(AP) ; ADJUST BY PARAMETER IN AST ARGLIST
      0249 608 RET ; AND RETURN
      0249 609
      0249 610 WSERR: BUG_CHECK WSSIZEERR,FATAL ; WORKING SET SIZE CALC IN ERROR
```

```
024D 612 .SBTTL SENDAST - Send AST to process
024D 613
024D 614 ++
024D 615 FUNCTIONAL DESCRIPTION: SENDAST IS CALLED BY SCH$QEND TO SEND ASTS TO THE
024D 616 PROCESS THAT INVOKE FUNCTIONS UNAVAILABLE TO THE ENVIRONMENT OF SCH$QEND.
024D 617 THESE INCLUDE ADJUSTING THE WORKING SET AND EXITTING.
024D 618
024D 619 INPUT PARAMETERS:
024D 620
024D 621 R0 - ADDRESS OF ACCESS MODE FOR AST
024D 622 (NEGATIVE CONTENTS PREVENT SENDING AST)
024D 623 R3 - AST PARAMETER
024D 624 R4 - PCB ADDRESS
024D 625 (SP) - AST ADDRESS
024D 626 4(SP) - RETURN ADDRESS FOR THIS SUBROUTINE
024D 627
024D 628 --
024D 629 SENDAST:
024D 630
024D 631 PUSHL R0 ; SAVE ADDRESS OF ACCESS MODE
024D 632 PUSHL R3 ; AND AST PARAMETER
024D 633 TSTB (R0) ; CHECK VALUE OF ACCESS MODE
024D 634 BLSS 10$ ; DO NOT QUEUE AST IF NEGATIVE
024D 635 BBS #PCB$V_DELPEN,PCB$S_STS(R4),10$ ; NOR IF MARKED FOR DELETE
024D 636 MOVZWL #ACB$C_LENGTH,R1 ; SET SIZE REQUIRED
024D 637 BSBW EXESALONONPAGED ; ALLOCATE A BLOCK
024D 638 BLBC R0,10$ ; NONE, TRY LATER
024D 639 MOVW #DYN$C_ACB,ACB$B_TYPE(R2) ; SET TYPE OF STRUCTURE
024D 640 MOVW R1,ACB$W_SIZE(R2) ; AND SIZE OF STRUCTURE
024D 641 MOVL (SP)+,ACB$S_ASTPRM(R2) ; AND AST PARAMETER VALUE
024D 642 MOVB @ (SP),ACB$B_RMOD(R2) ; SET ACCESS MODE FOR AST
024D 643 DECB @ (SP)+ ; INDICATE SUCCESS FOR THIS ACCESS MODE
024D 644 MOVL (SP)+,ACB$S_AST(R2) ; SET AST ADDRESS
024D 645 MOVL PCB$S_PID(R4),ACB$S_PID(R2) ; SET PID FOR AST
024D 646 PUSHR #*M<R4,R5> ; SAVE REGS FOR QAST
024D 647 MOVL R2,R5 ; SET ADDRESS OF ACB
024D 648 CLRL R2 ; NULL PRIORITY INCREMENT
024D 649 BSBW SCH$QAST ; QUEUE AST FOR PROCESS
024D 650 POPR #*M<R4,R5> ; RESTORE PCB,PHD ADDRESSES
024D 651 RSB ; EXIT
024D 652
024D 653 : Error path if nonpaged pool allocation fails or if AST access mode is
024D 654 : negative, indicating either an AST in progress (for automatic working
024D 655 : set adjustment) or all access modes are done (for CPU time limit expiration)
024D 656
024D 657 10$: ADDL #12,SP ; CLEAN PARAMETERS FROM STACK
024D 658 RSB ; AND EXIT
024D 659
024D 660
024D 661
024D 662
024D 663
024D 664
024D 665
024D 666
024D 667
024D 668
024D 669
024D 670
024D 671
024D 672
024D 673
024D 674
024D 675
024D 676
024D 677
024D 678
024D 679
024D 680
024D 681
024D 682
024D 683
024D 684
024D 685
024D 686
024D 687
024D 688
024D 689
024D 690
024D 691
024D 692
024D 693
024D 694
024D 695
024D 696
024D 697
024D 698
024D 699
024D 700
```

```
0290 658
0290 659 .SBTTL SCH$WAKE - WAKE PROCESS INTERNAL
0290 660 :++
0290 661 : FUNCTIONAL DESCRIPTION:
0290 662 : SCH$WAKE WAKES THE PROCESS SPECIFIED BY THE PID SUPPLIED.
0290 663 :
0290 664 : CALLING SEQUENCE:
0290 665 : BSB/JSB SCH$WAKE
0290 666 :
0290 667 : INPUT PARAMETERS:
0290 668 : R1 - PID OF PROCESS TO WAKE
0290 669 :
0290 670 : OUTPUT PARAMETERS:
0290 671 : R0 - COMPLETION STATUS CODE
0290 672 : R4 - PCB ADDRESS OF PROCESS AWAKENED
0290 673 :
0290 674 : COMPLETION CODES:
0290 675 : $$$_NORMAL - NORMAL SUCCESSFUL COMPLETION STATUS
0290 676 : $$$_NONEXPR - NONEXISTENT PROCESS (INVALID PID)
0290 677 :
0290 678 : ENVIRONMENT:
0290 679 : IPL = IPL$_SYNCH
0290 680 :
0290 681 :--
0290 682 : SCH$WAKE::
0290 683 : MOVZWL R1,R4 : WAKE PROCESS INTERNAL
0290 684 : MOVL @W^SCH$GL_PCBVEC[R4],R4 : GET PROCESS INDEX (PIX)
0290 685 : CMPL R1,PCB$_PID(R4) : LOOK UP PCB ADDRESS
0290 686 : BNEQ 30$ : VERIFY PID
0290 687 : BBSS #PCB$_WAKEPEN,PCB$_STS(R4),10$ : REPORT ERROR
0290 688 : 10$: : SET WAKE PENDING
0290 689 : MOVZBL #PRI$_RESAVL,R2 : SET PRIORITY INCREMENT CLASS
0290 690 : RPTEVT WAKE : REPORT WAKE EVENT
0290 691 : MOVZWL #$$$_NORMAL,R0 : SET SUCCESS CODE
0290 692 : RSB : RETURN
0290 693 : 20$:
0290 694 : 30$: MOVZWL #$$$_NONEXPR,R0 : SET NONEXISTENT PROCESS STATUS
0290 695 : RSB
```

54	54	51	3C	0290	683	MOVZWL	R1,R4	:	WAKE PROCESS INTERNAL
54	0000'DF44	D0	0293	684	MOVL	@W^SCH\$GL_PCBVEC[R4],R4	:	:	GET PROCESS INDEX (PIX)
60	A4	51	D1	0299	685	CMPL	R1,PCB\$_PID(R4)	:	LOOK UP PCB ADDRESS
		10	12	029D	686	BNEQ	30\$	:	VERIFY PID
00	24	A4	0C	E2	029F	687	BBSS	:	REPORT ERROR
				02A4	688	10\$:		:	SET WAKE PENDING
	52	02	9A	02A4	689	MOVZBL	#PRI\$_RESAVL,R2	:	SET PRIORITY INCREMENT CLASS
				02A7	690	RPTEVT	WAKE	:	REPORT WAKE EVENT
	50	01	3C	02AB	691	MOVZWL	#\$\$\$_NORMAL,R0	:	SET SUCCESS CODE
			05	02AE	692	RSB		:	RETURN
				02AF	693			:	
50	0BE8	8F	3C	02AF	694	30\$:	MOVZWL	:	SET NONEXISTENT PROCESS STATUS
			05	02B4	695	RSB		:	



```
02B5 697 .SBTTL SCH$SWPWAKE - WAKE SWAPPER PROCESS
02B5 698 :++
02B5 699 : FUNCTIONAL DESCRIPTION:
02B5 700 : SCH$SWPWAKE AWAKENS THE SWAPPER PROCESS TO PERFORM SOME OPERATION.
02B5 701 :
02B5 702 : CALLING SEQUENCE:
02B5 703 : BSB/JSB SCH$SWPWAKE
02B5 704 :
02B5 705 : INPUT PARAMETERS:
02B5 706 : NONE
02B5 707 :
02B5 708 : OUTPUT PARAMETERS:
02B5 709 : R0-R4 PRESERVED
02B5 710 :
02B5 711 : SIDE EFFECTS:
02B5 712 : A WAKE EVENT IS GENERATED FOR THE SWAPPER PROCESS WHICH CAN
02B5 713 : CAUSE THE PROCESSOR TO BE RESCHEDULED.
02B5 714 :
02B5 715 :--
02B5 716 :
02B5 717 SCH$SWPWAKE::
02B5 718 TSTL L^SCH$GL_COMOQS ; WAKE SWAPPER PROCESS
02B5 719 BNEQ 10$ ; ANY INSWAP CANDIDATES?
02B5 720 CMPL L^SCH$GL_MFYCNT,L^SCH$GL_MFYLM ; ARE THERE MODIFIED PAGES TO WRITE?
02B5 721 BGEQ 10$ ; YES, MUST WAKE SWAPPER THEN
02B5 722 CMPL L^SCH$GL_FREECNT,L^SCH$GL_FREEIM ; DO WE NEED FREE PAGES?
02B5 723 BLSS 10$ ; YES, MUST WAKE SWAPPER THEN
02B5 724 TSTW L^SCH$GW_DELPHDCT ; ARE THERE DELETED HEADERS TO PURGE?
02B5 725 BNEQ 10$ ; YES, MUST WAKE SWAPPER THEN
02B5 726 TSTL W^EX$GL_PFATIM ; WAS THERE A POWER FAIL RECOVERY?
02B5 727 BEQL 20$ ; BR IF NONE
02B5 728 10$: TSTB L^SCH$GB_SIP ; SWAPPER ALREADY BUSY?
02B5 729 BNEQ 20$ ; BR IF YES
02B5 730 PUSHR #^M<R0,R1,R2,R3,R4> ; SAVE R0-R4
02B5 731 MOVL W^SCH$GL_SWPPID,R1 ; GET PID OF SWAPPER
02B5 732 BSBB SCH$WAKE ; AND AWAKEN IT
02B5 733 POPR #^M<R0,R1,R2,R3,R4> ; RESTORE R0-R4
02B5 734 20$: RSB ; AND RETURN TO CALLER
02B5 735
02B5 736 .END
```

00000000'EF 05 02B5 718 TSTL L^SCH\$GL\_COMOQS ; WAKE SWAPPER PROCESS  
28 12 02B8 719 BNEQ 10\$ ; ANY INSWAP CANDIDATES?  
00000000'EF 01 02BD 720 CMPL L^SCH\$GL\_MFYCNT,L^SCH\$GL\_MFYLM ; ARE THERE MODIFIED PAGES TO WRITE?  
1B 18 02C8 721 BGEQ 10\$ ; YES, MUST WAKE SWAPPER THEN  
00000000'EF 01 02CA 722 CMPL L^SCH\$GL\_FREECNT,L^SCH\$GL\_FREEIM ; DO WE NEED FREE PAGES?  
0E 19 02D5 723 BLSS 10\$ ; YES, MUST WAKE SWAPPER THEN  
00000000'EF 05 02D7 724 TSTW L^SCH\$GW\_DELPHDCT ; ARE THERE DELETED HEADERS TO PURGE?  
06 12 02DD 725 BNEQ 10\$ ; YES, MUST WAKE SWAPPER THEN  
0000'CF 05 02DF 726 TSTL W^EX\$GL\_PFATIM ; WAS THERE A POWER FAIL RECOVERY?  
13 13 02E3 727 BEQL 20\$ ; BR IF NONE  
00000000'EF 95 02E5 728 10\$: TSTB L^SCH\$GB\_SIP ; SWAPPER ALREADY BUSY?  
0B 12 02EB 729 BNEQ 20\$ ; BR IF YES  
1F 0B 02ED 730 PUSHR #^M<R0,R1,R2,R3,R4> ; SAVE R0-R4  
51 0000'CF 00 02EF 731 MOVL W^SCH\$GL\_SWPPID,R1 ; GET PID OF SWAPPER  
9A 10 02F4 732 BSBB SCH\$WAKE ; AND AWAKEN IT  
1F 0A 02F6 733 POPR #^M<R0,R1,R2,R3,R4> ; RESTORE R0-R4  
05 05 02F8 734 20\$: RSB ; AND RETURN TO CALLER  
02F9 735  
02F9 736 .END

Variable	Value	Attribute	Value
PCBSW_STATE	= 0000002C		
PHDSB_AWSMODE	= 00000061		
PHDSB_CPUMODE	= 00000060		
PHDSL_CPULIM	= 0000005C		
PHDSL_CPUTIM	= 00000038		
PHDSL_EXTRACPU	= 00000110		
PHDSL_PAGEFLTS	= 0000004C		
PHDSL_PC	= 000000C0		
PHDSL_PFLREF	= 000000FC		
PHDSL_PFLRATE	= 000000F8		
PHDSL_RO	= 00000088		
PHDSL_TIMREF	= 00000100		
PHDSW_QUANT	= 0000003C		
PHDSW_WSEXTENT	= 00000016		
PHDSW_WSLIST	= 00000008		
PHDSW_WSQUOTA	= 00000018		
PHDSW_WSSIZE	= 00000050		
PRS_IPL	= 00000012		
PRS_SIRR	= 00000014		
PRIS_RESAVL	= 00000002		
RSE...	= 00000034	R	03
SCH\$AQ_COMT	*****	X	03
SCH\$AQ_WQHDR	*****	X	03
SCH\$CHSE	00000088	RG	03
SCH\$CHSEP	000000A4	RG	03
SCH\$C_CEF	= 00000003		
SCH\$C_COLPG	= 00000001		
SCH\$C_COM	= 0000000C		
SCH\$C_COMO	= 0000000D		
SCH\$C_FPG	= 0000000B		
SCH\$C_HIB	= 00000007		
SCH\$C_HIBO	= 00000008		
SCH\$C_LEF	= 00000005		
SCH\$C_LEFO	= 00000006		
SCH\$C_MWAIT	= 00000002		
SCH\$C_PFW	= 00000004		
SCH\$C_SUSP	= 00000009		
SCH\$C_SUSPO	= 0000000A		
SCH\$FORCEDEXIT	0000018D	RG	03
SCH\$GB_PRI	*****	X	03
SCH\$GB_SIP	*****	X	03
SCH\$GL_AWSTIME	*****	X	03
SCH\$GL_BORROWLIM	*****	X	03
SCH\$GL_COMOQS	*****	X	03
SCH\$GL_COMQS	*****	X	03
SCH\$GL_FREECNT	*****	X	03
SCH\$GL_FREELIM	*****	X	03
SCH\$GL_MFYCNT	*****	X	03
SCH\$GL_MFY LIM	*****	X	03
SCH\$GL_PCBVEC	*****	X	03
SCH\$GL_PFRATH	*****	X	03
SCH\$GL_PFRATL	*****	X	03
SCH\$GL_SWPPID	*****	X	03
SCH\$GL_WSDEC	*****	X	03
SCH\$GL_WSINC	*****	X	03
SCH\$GW_AWSMIN	*****	X	03
SCH\$GW_DELPDCT	*****	X	03

RSE  
Symbol table

- REPORT SYSTEM EVENT

L 3

16-SEP-1984 01:06:34 VAX/VMS Macro V04-00  
5-SEP-1984 03:47:04 [SYS.SRC]RSE.MAR;1

Page 20  
(1)

SCH\$GW QUAN	*****	X	03
SCH\$QAST	*****	X	03
SCH\$QEND	0000012A	RG	03
SCH\$RSE	00000000	RG	03
SCH\$SWPWAKE	000002B5	RG	03
SCH\$UNWAIT	0000004D	RG	03
SCH\$WAKE	00000290	RG	03
SENDAST	0000024D	R	03
SGN\$GL_EXTRACPU	*****	X	03
SS\$_EXCPUTIM	= 000020AC		
SS\$_NONEXPR	= 000008E8		
SS\$_NORMAL	= 00000001		
ST	= 00000002		
STACT	0000001E	R	03
STET	= 00000000	R	02
STMSK	= 00001000		
SWPO	00000112	R	03
SWPOE	00000123	R	03
SY\$ADJWSL	*****	GX	03
SY\$EXIT	*****	GX	03
WAITMSK	0000007B	R	03
WAITST	= 00000FFE		
WQH\$C_LENGTH	= 0000000C		
WQH\$C_WQBL	= 00000004		
WQH\$C_WQCN	= 00000008		
WSADJUST	000001A0	R	03
WSERR	00000249	R	03

+-----+  
! Psect synopsis !  
+-----+

PSECT name

Allocation	PSECT No.	Attributes
00000000 ( 0.)	00 ( 0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
00000000 ( 0.)	01 ( 1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
0000002C ( 44.)	02 ( 2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
000002F9 ( 761.)	03 ( 3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

+-----+  
! Performance indicators !  
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	36	00:00:00.09	00:00:01.96
Command processing	123	00:00:00.48	00:00:05.87
Pass 1	331	00:00:10.62	00:00:34.14
Symbol table sort	0	00:00:01.68	00:00:04.35
Pass 2	140	00:00:02.57	00:00:09.68
Symbol table output	18	00:00:00.13	00:00:00.13
Psect synopsis output	2	00:00:00.02	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	652	00:00:15.60	00:00:56.16

The working set limit was 1650 pages.  
62765 bytes (123 pages) of virtual memory were used to buffer the intermediate code.



There were 60 pages of symbol table space allocated to hold 1060 non-local and 26 local symbols.  
736 source lines were read in Pass 1, producing 19 object records in Pass 2.  
26 pages of virtual memory were used to define 25 macros.

-----  
! Macro library statistics !  
-----

Macro library name	Macros defined
-----	-----
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	12
\$255\$DUA28:[SYS.LIB]STARLET.MLB;2	8
TOTALS (all libraries)	20

1119 GETS were required to define 20 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:RSE/OBJ=OBJ\$:RSE MSRC\$:RSE/UPDATE=(ENH\$:RSE)+EXECML\$/LIB



0380

AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

SHELL  
LIS

RSE  
LIS

SCBVECTOR  
LIS

SDAT  
LIS

SHMSDRTN  
LIS

SCSVEC  
LIS

SCHED  
LIS

SECAUDIT  
LIS

SPTSKE  
LIS

RUESVSUC  
LIS